Lab: Relation Extraction

顏安孜

azyen@cs.nycu.edu.tw

Briefly Introduction of Relation Extraction

- International Business Machines Corporation (IBM or the company) was incorporated in the State of New York on June 16, 1911, as the Computing-Tabulating-Recording Co. (C-T-R)…

Company-Founding	
Company	IBM
Location	New York
Date	June 16, 1911
Original-Name	Computing-Tabulating-Recording Co.

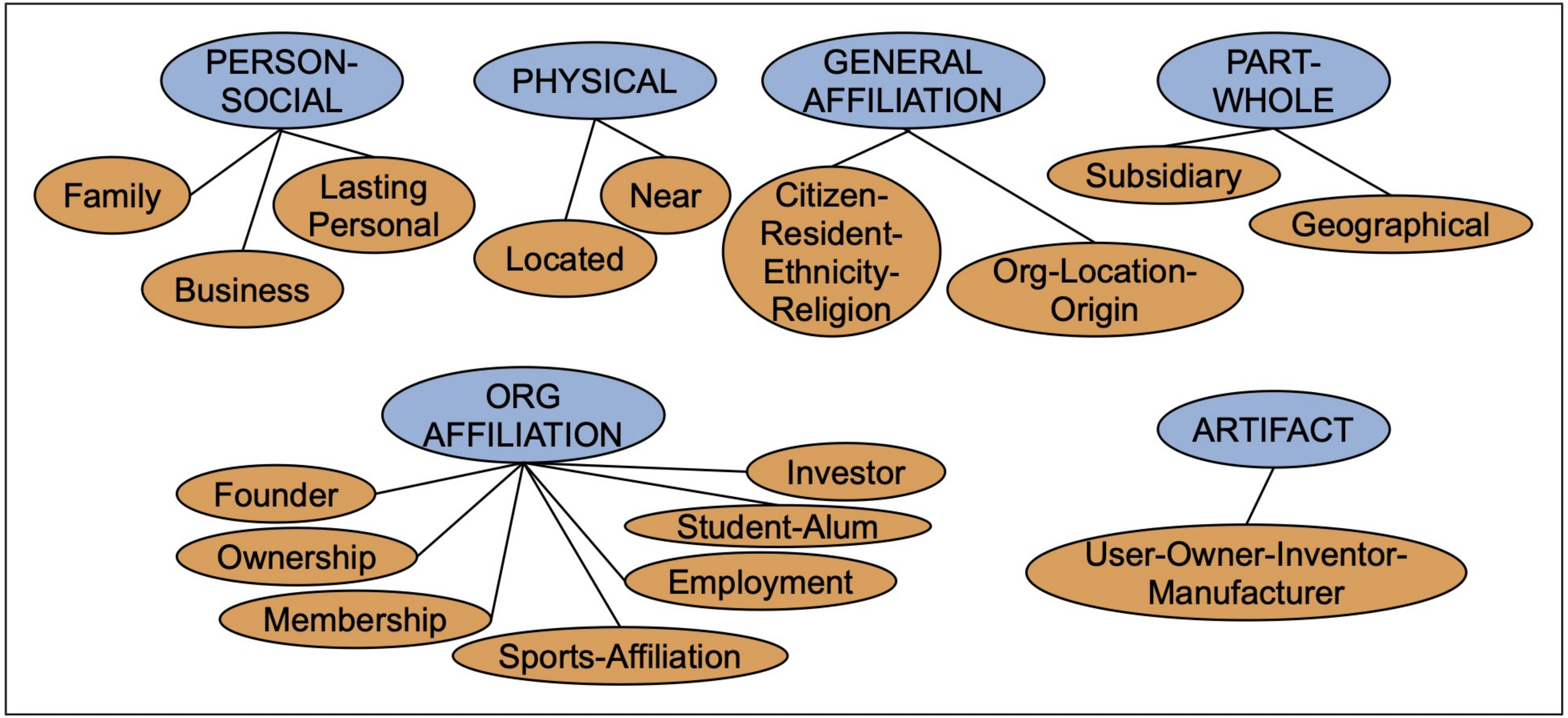
Founding-year(IBM,1911)
Founding-location(IBM, New York)
…

Why Relation Extraction

- Extract structure knowledge from text
    - Store world knowledge into knowledge bases
- Support question answering
    - (acted-in ?x “E.T.”)(is-a ?y actor)(granddaughter-of ?x ?y)

- But which relations should we extract?

Automated Content Extraction (ACE)



Automated Content Extraction (ACE)

- Physical-Located PER-GPE

  He was in Taiwan

- Part-Whole-Subsidiary ORG-ORG

  XYZ, the parent company of ABC

- Person-Social-Family PER-PER

  John's wife Yoko

- Org-AFF-Founder PER-ORG

  Steve Jobs, co-founder of Apple…

Medical Resource: UMLS (Unified Medical Language System)

Entity	Relation	Entity
Injury	disrupts	Physiological Function
Bodily Location	location-of	Biologic Function
Anatomical Structure	part-of	Organism
Pharmacologic Substance	causes	Pathological Function
Pharmacologic Substance	treats	Pathologic Function

Doppler echocardiography can be used to diagnose left anterior descending artery stenosis in patients with type 2 diabetes

➔ *Echocardiography, Doppler* Diagnoses *Acquired stenosis*

Extracting Relation Triples from Text

National Yang Ming Chiao Tung University

From Wikipedia, the free encyclopedia

National Yang Ming Chiao Tung University (**NYCU**;[1][2] Chinese: 國立陽明交通大學) is a public research university in Taiwan. It was created in 2021 through the merger of National Yang-Ming University and National Chiao Tung University. At present, there are 19 colleges, 74 university/college level research centers and 1 hospital in Yilan. The university is one of the four universities selected by the Ministry of Education to participate in the Global Connecting Whole University Program.[3][4]

National Yang Ming Chiao Tung University

國立陽明交通大學	
Type	Public
Established	2021; 1 year ago
Principal	Lin Chi-hung
Academic staff	1,152 (2019)
Students	19,078 (2019)
Location	Hsinchu City and Taipei, Taiwan
Affiliations	University System of Taiwan
Website	www.nycu.edu.tw/en/

Contents [hide]

National Yang Ming Chiao Tung University	
Simplified Chinese	国立阳明交通大学
Traditional Chinese	國立陽明交通大學
Transcriptions	[show]

History [edit]

See also: National Yang-Ming University § History, and National Chiao Tung University § History

- Triples:
- NYCU, is_a, research university
- NYCU, located_in, Taiwan
- NYCU, was_created_in, 2021

Databases of Wikipedia Relations

National Yang Ming Chiao Tung University

國立陽明交通大學

Type	Public
Established	2021; 1 year ago
Principal	Lin Chi-hung
Academic staff	1,152 (2019)
Students	19,078 (2019)
Location	Hsinchu City and Taipei, Taiwan
Affiliations	University System of Taiwan
Website	www.nycu.edu.tw/en/

國立陽明交通大學

NATIONAL YANG MING CHIAO TUNG UNIVERSITY

- NYCU, type, public
- NYCU, Established, 2021
- NYCU, Principal, Lin Chi-hung
- NYCU, Academic staff, 1,152 (2019)
- NYCU, Students, 19,078 (2019)
- NYCU, Location, Hsinchu City and Taipei, Taiwan

Ontological Relations

- WordNet
- IS-A (hypernym): subsumption between classes
  - Giraffe IS-A ruminant(反芻動物) IS-A ungulate IS-A mammal(有蹄類) IS-A vertebrate(脊椎動物) IS-A animal…
- Instance-of: relation between individual and class
  - San Francisco instance-of city

How to extract relation?

1. Hand-written patterns
2. Supervised machine learning
3. Semi-supervised and unsupervised
   - Bootstrapping (using seeds)
   - Distant supervision
   - Unsupervised learning from the web

The process we may need

- Named Entity Recognition
- Coreference Resolution
- Dependency Paring

Named Entity Recognition

- Named Entity is anything that can be referred to with a proper name: a person, a location, an organization

Word	POS	Entity tag
Janet	noun	person
NYCU	noun	organization
Hsinchu	noun	location

- The task of named entity recognition (NER) is to find spans of text that constitute proper names and tag the type of the entity

Entity Tags

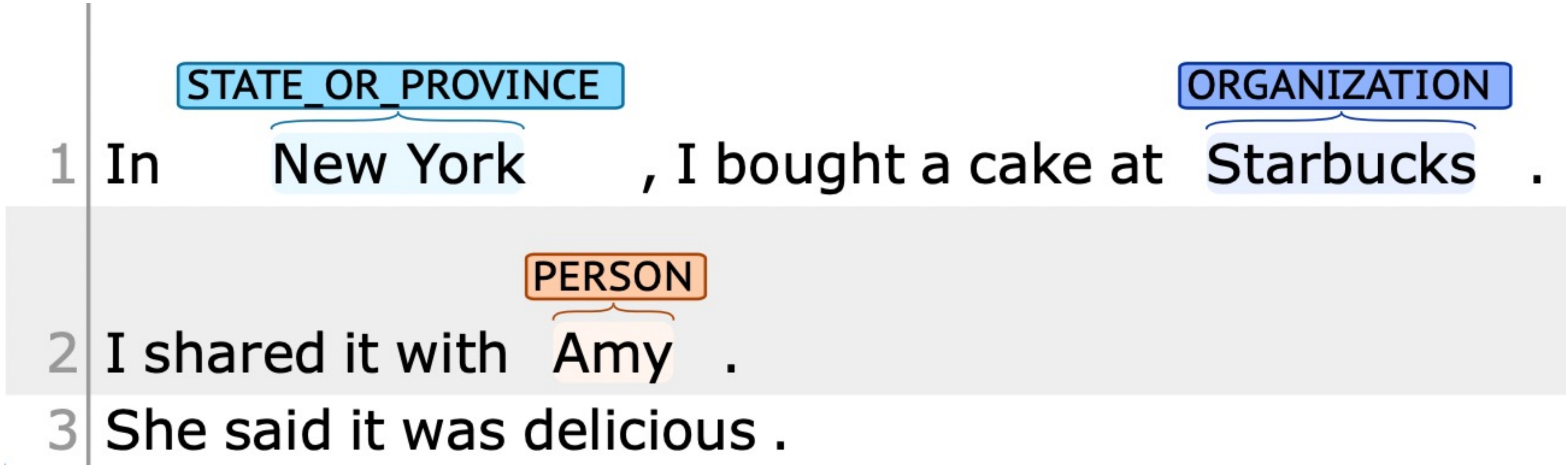
- Most common entity tags : PER (person), LOC (location), ORG (organization), GPE (geo-political entity), TIME(times), …

Type	Tag	Sample Categories	Example sentences
People	PER	people, characters	**Turing** is a giant of computer science.
Organization	ORG	companies, sports teams	The **IPCC** warned about the cyclone.
Location	LOC	regions, mountains, seas	**Mt. Sanitas** is in **Sunshine Canyon**.
Geo-Political Entity	GPE	countries, states	**Palo Alto** is raising the fees for parking.

- Type ambiguities

[$_{\text{PER}}$ Washington] was born into slavery on the farm of James Burroughs.
[$_{\text{ORG}}$ Washington] went up 2 games to 1 in the four-game series.
Blair arrived in [$_{\text{LOC}}$ Washington] for what may well be his last state visit.
In June, [$_{\text{GPE}}$ Washington] passed a primary seatbelt law.

In New York, I bought a cake at Starbucks. I shared it with Amy. She said it was delicious.



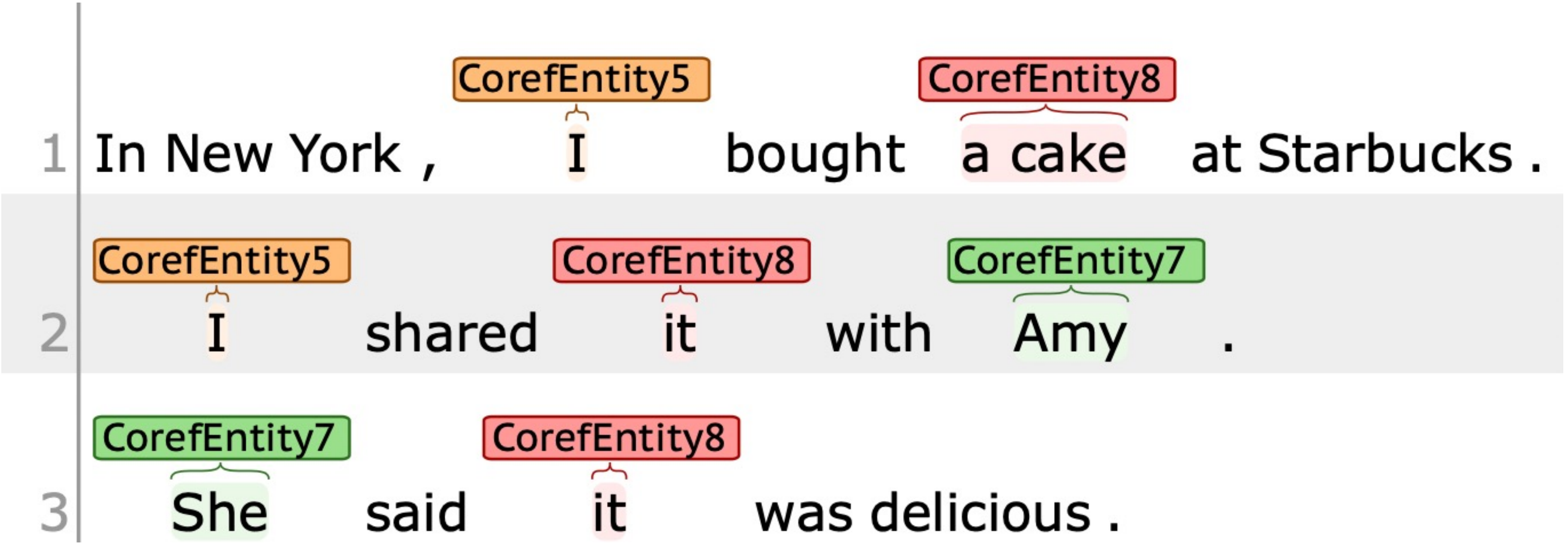
https://corenlp.run

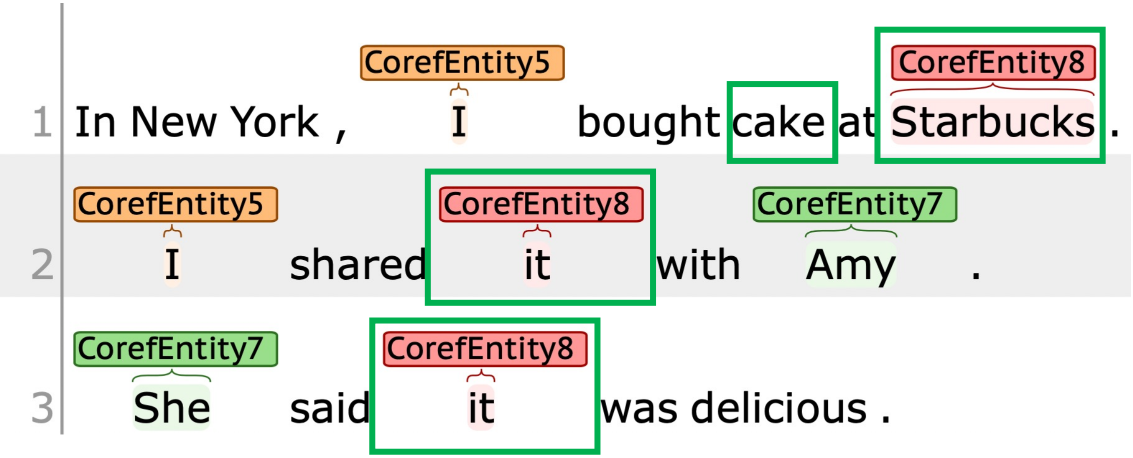
Coreference Resolution

- Identifying mentions in text that refer to the same underlying real world entities

- Example:
  - <u>Victoria Chen</u>, CFO of Megabucks Banking, saw <u>her</u> pay jump to $2.3 million, as <u>the 38-year-old</u> became the company's president.

In New York, I bought a cake at Starbucks. I shared it with Amy. She said it was delicious.

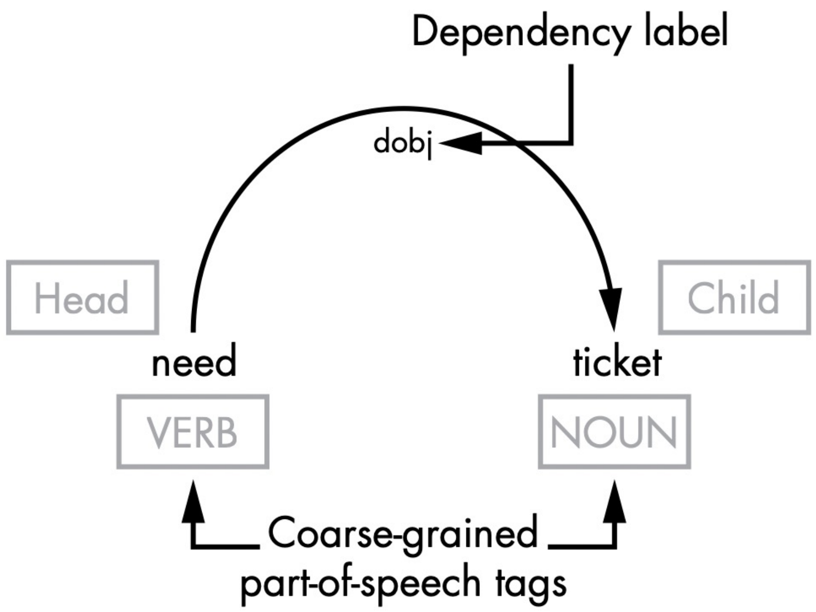


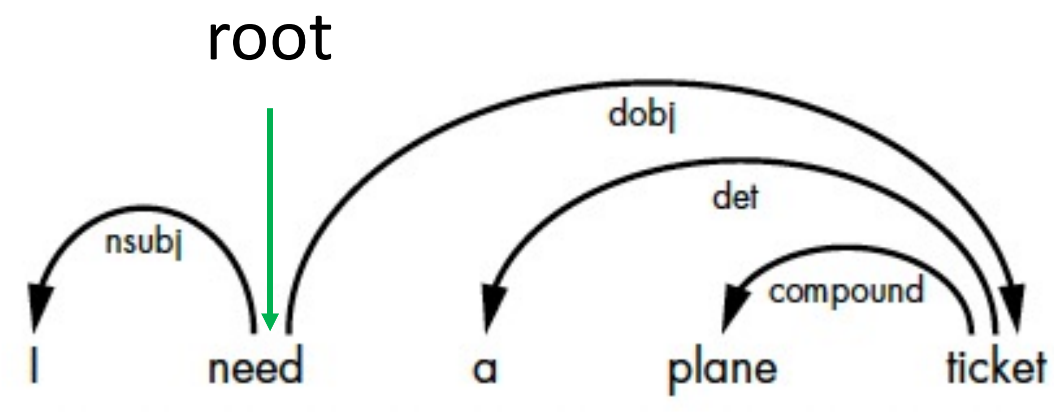


Without the determiner “a”

Dependency Parsing

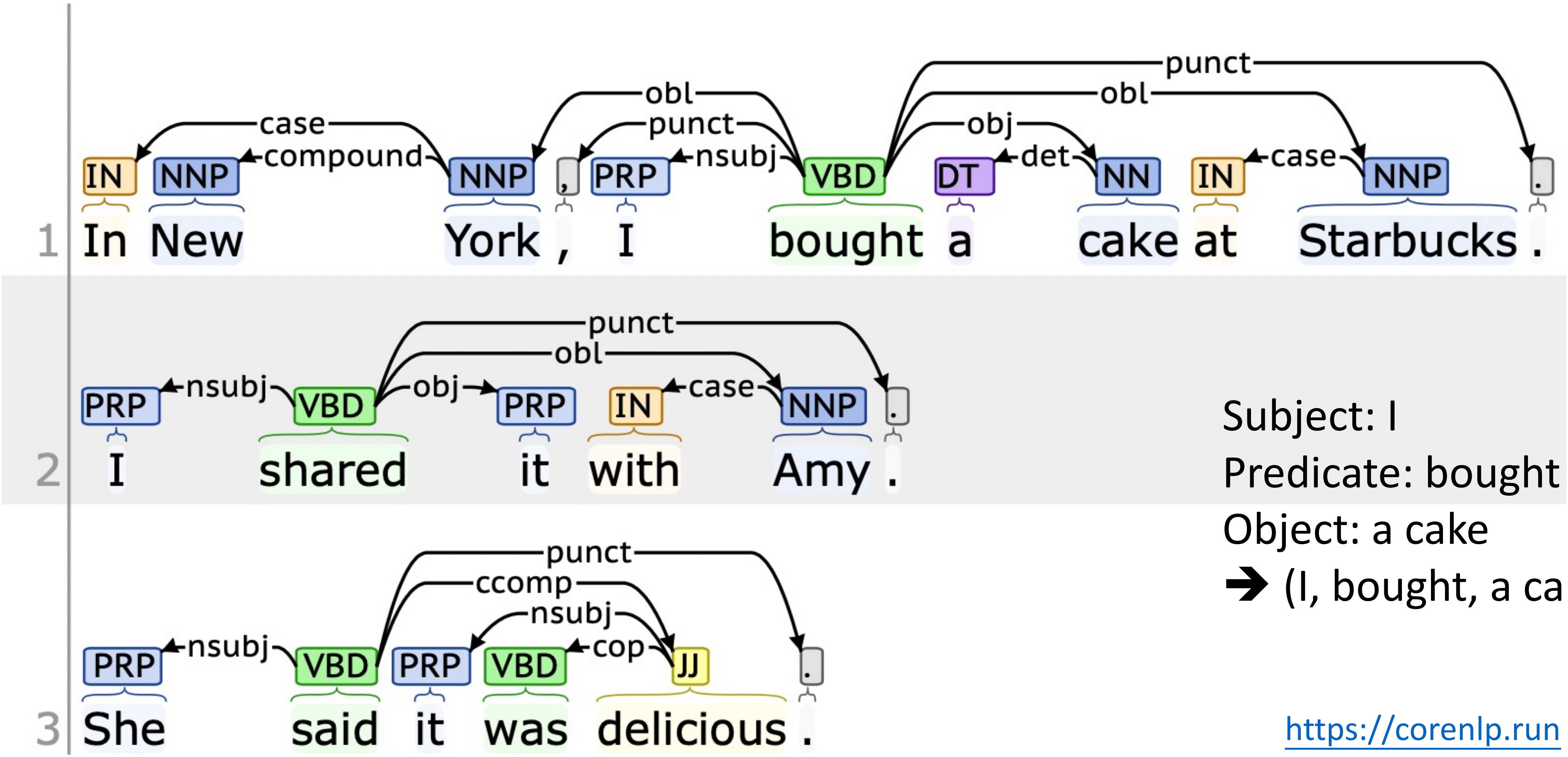
- Dependency parse shows dependencies in a sentence
- Dependency = the type of syntactic relation between two words in a sentence





Reference: Natural Language Processing with Python and spaCy by Yuli Vasiliev

In New York, I bought a cake at Starbucks. I shared it with Amy. She said it was delicious.



Subject: I
Predicate: bought
Object: a cake
➔ (I, bought, a cake)

https://corenlp.run

Dependency Relations from the Universal Dependency

Clausal Argument Relations	Description
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction

```
>>> import spacy
>>> spacy.explain('nsubj')
'nominal subject'
>>> spacy.explain('pobj')
'object of preposition'
```

https://universaldependencies.org/docs/en/dep/

Application: Task-Oriented Dialogue Systems

1 • Domain Classification

2 • Intent Classification

3 • Slot Filling

- Reserving restaurant
- Booking a movie ticket
- Booking a flight ticket

I want to buy a ticket to Japan this Sunday.

➔ Intent = Buy_ticket

Flight DB

SELECT flight{
flt.dest = “Japan”,
flt.date = “3/5”}

Origin	Destination	Date
Korean	Japan	3/5
Taiwan	Japan	3/6
…	…	…

Example of Extracting Entities and Relations

Extracting Entities

```python
sent = 'I need a ticket to Los Angeles on May 8th.'
```

```python
doc = nlp(sent)

for token in doc:
    if token.ent_type != 0:
        print(token.text, token.ent_type_)
```

```
Los GPE
Angeles GPE
May DATE
8th DATE
```

Processing a customer's request
Extracting necessary information

Extracting Noun Chunks

```python
ner_s = 'Apple investors urged to vote against a nearly \
$100 million pay package for CEO Tim Cook.'

doc = nlp(ner_s)

for token in doc:
    if token.ent_type != 0:
        print(token.text, token.ent_type_)
```

```
Apple ORG
nearly MONEY
$ MONEY
100 MONEY
million MONEY
Tim PERSON
Cook PERSON
```

```python
for noun_chunk in doc.noun_chunks:
    print(noun_chunk.text)
```

```
Apple investors
a nearly $100 million pay package
CEO
Tim Cook
```

You can also train your own NER model with spaCy ☺

Extracting Relations

```python
import spacy
```

```python
nlp = spacy.load("en_core_web_sm")
```

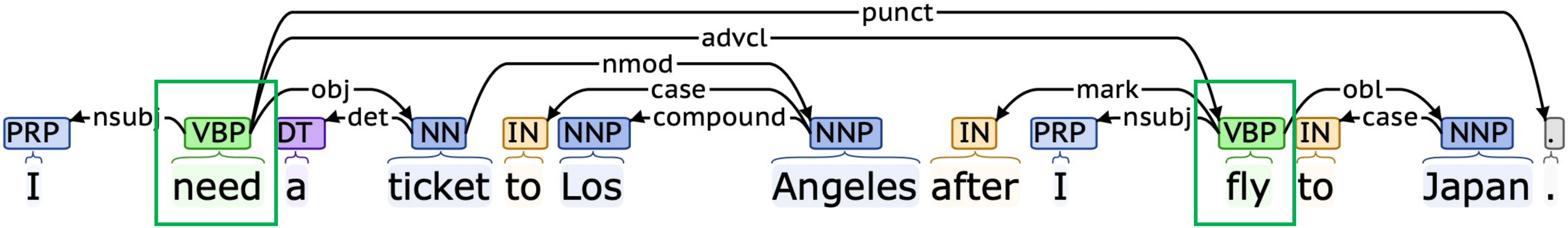
```python
sent = 'I need a ticket to Los Angeles on May 8th.'
```

```python
doc = nlp(sent)

for token in doc:
    print('%s\t%s\t%s\t%s\t%s' %(token.text, token.pos_, token.dep_,
                                 spacy.explain(token.dep_), token.head.text))
```

```
I       PRON    nsubj   nominal subject need
need    VERB    ROOT    None    need
a       DET     det     determiner      ticket
ticket  NOUN    dobj    direct object   need
to      ADP     prep    prepositional modifier  ticket
Los     PROPN   compound        compound        Angeles
Angeles PROPN   pobj    object of preposition   to
on      ADP     prep    prepositional modifier  need
May     PROPN   compound        compound        8th
8th     NOUN    pobj    object of preposition   on
.       PUNCT   punct   punctuation     need
```

More Complicated Syntaxis Structure



- POS tags is beneficial for extracting information in a more complicated sentence

(I, *need*, a ticket)

(I, *fly*, Japan)

Extracting triples from the sentence

```python
sentence = 'I established my own workshop in 2018 before I went to Japan.'
doc = nlp(sentence)
for token in doc:
    print('%s\t%s\t%s\t%s\t%s\t%d' %(token.text, token.pos_, token.dep_, \
                                    spacy.explain(token.dep_), token.head.text, token.head.i))
```

```
I	PRON	nsubj	nominal subject	established	1
established	VERB	ROOT	None	established	1
my	PRON	poss	possession modifier	workshop	4
own	ADJ	amod	adjectival modifier	workshop	4
workshop	NOUN	dobj	direct object	established	1
in	ADP	prep	prepositional modifier	established	1
2018	NUM	pobj	object of preposition	in	5
before	SCONJ	mark	marker	went	9
I	PRON	nsubj	nominal subject	went	9
went	VERB	advcl	adverbial clause modifier	established	1
to	ADP	prep	prepositional modifier	went	9
Japan	PROPN	pobj	object of preposition	to	10
.	PUNCT	punct	punctuation	established	1
```

```
I         PRON     nsubj    nominal subject established     1
established        VERB     ROOT     None     established     1
my        PRON     poss     possession modifier     workshop        4
own       ADJ      amod     adjectival modifier     workshop        4
workshop           NOUN     dobj     direct object   established     1
in        ADP      prep     prepositional modifier  established     1
2018      NUM      pobj     object of preposition   in      5
before    SCONJ    mark     marker   went     9
I         PRON     nsubj    nominal subject went     9
went      VERB     advcl    adverbial clause modifier       established     1
to        ADP      prep     prepositional modifier  went     9
Japan     PROPN    pobj     object of preposition   to      10
.         PUNCT    punct    punctuation     established     1
```

```python
for token in doc:
    subtree = list(token.subtree)
    print(token, subtree)
```

```
I [I]
established [I, established, my, own, workshop, in, 2018, before, I, went, to, Japan, .]
my [my]
own [own]
workshop [my, own, workshop]
in [in, 2018]
2018 [2018]
before [before]
I [I]
went [before, I, went, to, Japan]
to [to, Japan]
Japan [Japan]
. [.]
```

```python
verb_idxs = [(i, token) for i, token in enumerate(doc) if token.pos_ == 'VERB']
print(verb_idxs)
```

```
[(1, established), (9, went)]
```

```
I          PRON    nsubj   nominal subject established     1
established        VERB    ROOT    None    established     1
my         PRON    poss    possession modifier     workshop        4
own        ADJ     amod    adjectival modifier     workshop        4
workshop           NOUN    dobj    direct object   established     1
in         ADP     prep    prepositional modifier  established     1
2018       NUM     pobj    object of preposition   in      5
before     SCONJ   mark    marker  went    9
I          PRON    nsubj   nominal subject went    9
went       VERB    advcl   adverbial clause modifier       established     1
to         ADP     prep    prepositional modifier  went    9
Japan      PROPN   pobj    object of preposition   to      10
.          PUNCT   punct   punctuation     established     1
```

```python
def get_phrase(doc, head_idx, tag):
    for token in doc:
        if tag in token.dep_ and token.head.i == head_idx:
            subtree = list(token.subtree)
            start = subtree[0].i
            end = subtree[-1].i + 1
            return doc[start:end]
```

```python
doc = nlp(sentence)

for verb_idx in verb_idxs:
    subject_phrase = get_phrase(doc, verb_idx[0], 'subj')
    object_phrase = get_phrase(doc, verb_idx[0], 'obj')
    print('subject:', subject_phrase)
    print('predicate:', doc[verb_idx[0]])
    print('object:', object_phrase)
```

```
subject: I
predicate: established
object: my own workshop
subject: I
predicate: went
object: None
```

😯

- We have to design a better rule

```python
for token in doc:
    subtree = list(token.subtree)
    print(token, subtree)
```

```
I [I]
established [I, established, my, own,
my [my]
own [own]
workshop [my, own, workshop]
in [in, 2018]
2018 [2018]
before [before]
I [I]
went [before, I, went, to, Japan]
to [to, Japan]
Japan [Japan]
. [.]
```

CKIP Tagger

- 超級(A)　1000(Neu)　系列(Na)　全(Neqa)　英(Nc)　公開賽(Na)　將(D)　於(P)　3月(Nd)　16日(Nd)　登場(VA)　，(COMMACATEGORY)　昨(Nd)　（(PARENTHESISCATEGORY)　22日(Nd)　）(PARENTHESISCATEGORY)　籤表(Na)　出爐(VH)　，(COMMACATEGORY)　我國(Nc)　世界(Nc)　球后(Na)　戴資穎(Nb)　仍(D)　以(P)　第一(Neu)　種子(Na)　出戰(VC)　，(COMMACATEGORY)　尋求(VC)　個人(Nh)　在(P)　全(Neqa)　英(Nc)　公開賽(Na)　的(DE)　第四(Neu)　座(Nf)　冠軍(Na)　。(PERIODCATEGORY)　印度(Nc)　媒體(Na)　《(PARENTHESISCATEGORY)　滾動(VA)　》(PARENTHESISCATEGORY)　則(D)　報導(VE)　，(COMMACATEGORY)　「(PARENTHESISCATEGORY)　小戴(Nb)　」(PARENTHESISCATEGORY)　籤運(Na)　不錯(VH)　，(COMMACATEGORY)　如果(Cbb)　能(D)　穩定(VHC)　發揮(VJ)　，(COMMACATEGORY)　晉級(VJ)　八(Neu)　強(Na)　不(D)　是(SHI)　問題(Na)　。(PERIODCATEGORY)

https://colab.research.google.com/drive/1lafVJM6wKiRLH21qEbZ4qqpgkHErD8NW?usp=sharing

Appendix

Identifying Relations for Open Information Extraction

Anthony Fader, Stephen Soderland, and Oren Etzioni
University of Washington, Seattle
{afader,soderlan,etzioni}@cs.washington.edu

- EMNLP 2011

Extraction Algorithm of REVERB

- Relation Extraction: For each verb *v* in *s*, find the longest sequence of words r_v such that (1) r_v starts at *v*, (2) r_v satisfies the syntactic constraint, and (3) r_v satisfies the lexical constraint. If any pair of matches are adjacent or overlap in *s*, merge them into a single match.
- Argument Extraction: For each relation phrase *r* identified in Step 1, find the nearest noun phrase *x* to the left of *r* in s such that *x* is not a relative pronoun, WHO-adverb, or existential "there". Find the nearest noun phrase y to the right of r in s. If such an (*x*, *y*) pair could be found, return (*x*, *r*, *y*) as an extraction.

Hudson was born in Hampstead, which is a suburb of London.

e1: *(Hudson, was born in, Hampstead)*

e2: *(Hampstead, is a suburb of, London)*.

You Write Like You Eat: Stylistic variation as a predictor of social stratification

Angelo Basile
Symanto Research
Nürnberg, Germany
angelo.basile@symanto.net

Albert Gatt
University of Malta
Msida, Malta
albert.gatt@um.edu.mt

Malvina Nissim
University of Groningen
Groningen, The Netherlands
m.nissim@rug.nl

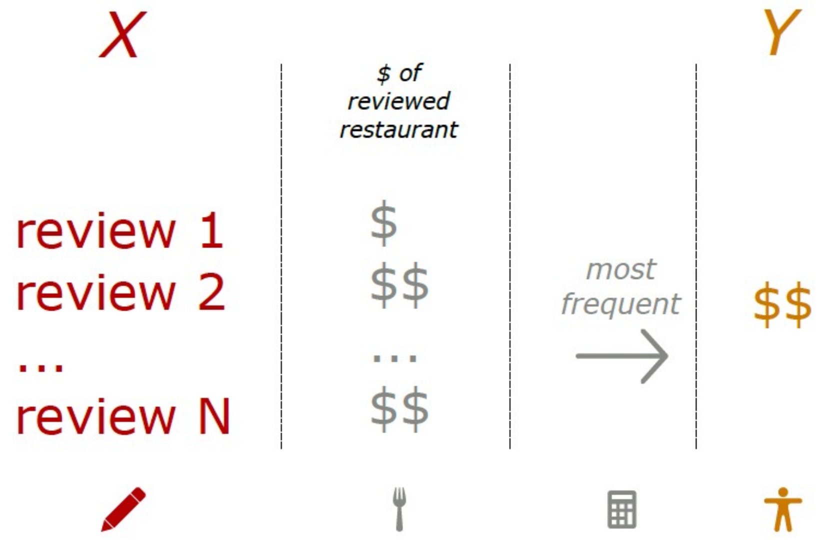
- ACL 2019

Motivation

- To what extent can socio-economic status be predicted from a person's text
- Can socio-economic groups be differentiated on the basis of syntactic features, compared to lexical features

➔ Collecting user-generated reviews labelled with a approximation of the socio-economic status of their author, based on the price range of restaurants

Four classes {$,$$,$$$,$$$$}



ranging across four classes related to increasing price

Sample reviews for classes $ and $$$$

CLASS $	CLASS $$$$
So freaking good. That's all I'm gonna say. Don't believe me? Walk into the place and smell it. [...] Will definitely go back.,Fresh, hand-made pepperoni rolls..... oh yeah. Their cheesy focattia (did I spell that right?) is amazing. Take it home, throw it in the oven, drizzle a little EVOO on top and you're golden. Friendly people there. Parking sucks, but I'm not taking off a point for that! Their marinara is dee-lish,Super tasty!!!	Let me start off saying that 2 years ago my husband and I had a spectacular dinner at L'Atelier by Joel Robuchon and finally got the "Time" to visit Joel Robuchon.We got a limo service and a nice tour inside the mansion of Robuchon which was very memorable and the hostess escorted us to the dining area. Decore: In comparison to L'Atelier this place was much more chic and elegant. However, I still loved the idea to see all the chefs preparing and decorating my plates at L'Atelier.

Analysis

$	$$	$$$	$$$$
fast	tried	at	excellent
kids	happy	clubs	gras
coffee	staff	wynn	we
customer	won	music	las
clean	put	pretty	steak
they	phoenix	night	tasting
order	find	club	foie
came	try	vegas	wine
always	place	buffet	course
pizza	salsa	hotel	vega

- Logistic Regression model

➔ The 10 most important word features per class

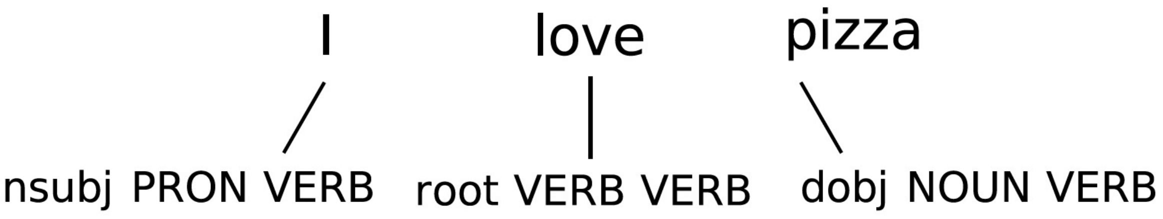
- Capture author-related stylistic features requires an abstraction away from the lexicon

Capturing Style

token	bleached representation
I	`X_01_True_V_2117`
really	`xxxxxx_06_True_CCVVCC_81`
love	`xxxx_06_True_CVCVCC_15`
pizza	`xxxxx_04_True_CCVC_617`
!	`!_01_False_!_21`

- Preserve the surface structure
  - Get rid of most lexical information

- Remove words and replace them with POS tags
  - Use dependency trees and expand the POS tags



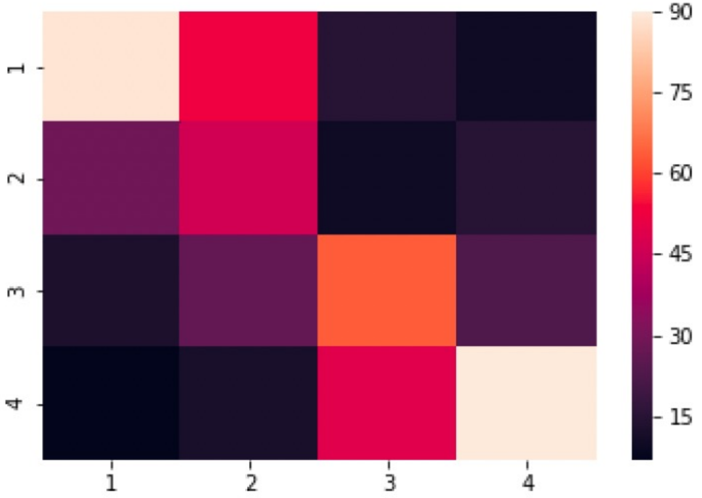
Experimental Results

model	F1
random baseline	0.25
LR BOW (lexical) baseline	0.53
CNN lexical	$\boxed{0.54}$
CNN pos tags	0.33
CNN dependency tree	$\boxed{0.52}$
CNN bleaching	0.46

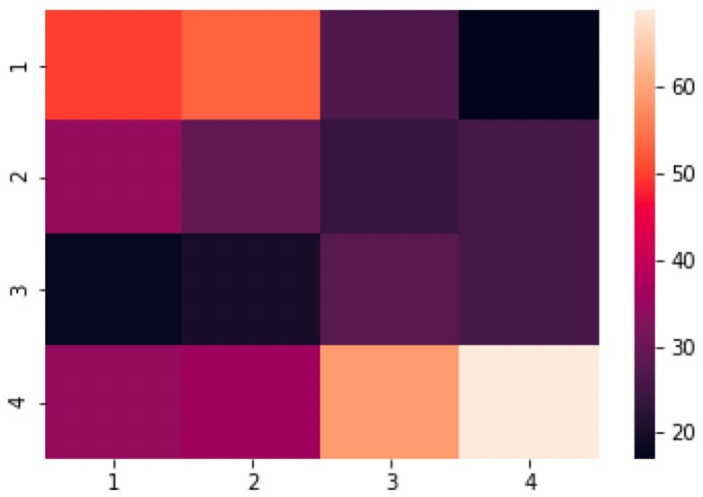
- It is possible to differentiate among the resulting classes both on the basis of type of establishment and on the basis of stylistic features in the writing style of its patrons

Confusion Matrices for the CNN Models

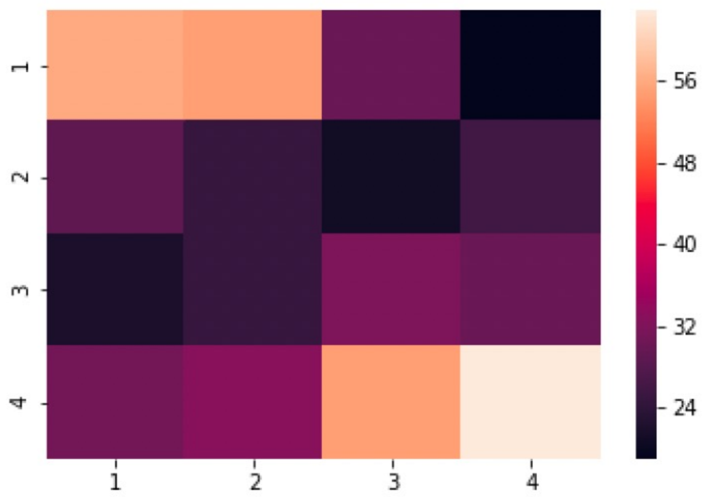
$	$$	$$$	$$$$
fast	tried	at	excellent
kids	happy	clubs	gras
coffee	staff	wynn	we
customer	won	music	las
clean	put	pretty	steak
they	phoenix	night	tasting
order	find	club	foie
came	try	vegas	wine
always	place	buffet	course
pizza	salsa	hotel	vega



(a) bleaching



(b) POS



(c) dependency trees

Reference

- Speech and Language Processing
  - Dan Jurafsky and James H. Martin

- Natural Language Processing with Python and spaCy by Yuli Vasiliev